Itô Bridge - Continuous Ant Colony Optimization (IB-CACO)

[1]Aldo Taranto
ORCID: 0000-0001-6763-4997
Aldo.Taranto@anu.edu.au
School of Computing
Australian National University
Canberra, 2601, ACT, Australia

[2]Ron Addie
ORCID: 0000-0002-6664-8462
Ron.Addie@unisq.edu.au
School of Mathematics, Physics and Computing
University of Southern Queensland
Toowoomba, 4350, QLD, Australia

November 16, 2025

***Abstract*—We advance the study of optimization over loss landscape surfaces (LLS), with particular emphasis on the high-dimensional and nonconvex settings characteristic of modern machine learning (ML). Building on our earlier work that introduced the Brownian Bridge–Continuous Ant Colony Optimization (BB-CACO) algorithm, we extend the approach to a more general framework based on Itô bridges. The resulting algorithm, Itô Bridge–CACO (IB-CACO), dynamically adjusts drift and diffusion parameters to enhance exploration and convergence. Unlike stochastic gradient descent (SGD) and its common variants (e.g., RMSProp, Adam), which are often hindered by noise, sensitivity to hyperparameters, and entrapment in local minima, IB-CACO consistently demonstrates superior performance in both locating global minima and achieving computational efficiency. Beyond ML benchmarks and training of Large language models (LLMs), the algorithm shows promise in domains where rapid and reliable global optimization is critical, including aircraft design, natural resource prospecting, drone navigation, and autonomous rescue operations.**

***Index Terms*—Fractional Brownian motion (fBm), Machine learning (ML), Algorithms, Ant colony optimization (ACO), Loss landscape surfaces (LLS).**

I. Introduction

Our survey paper [48] examined the literature on loss landscape surfaces (LLS), their nature and how stochastic gradient descent (SGD), despite all of its brilliance and advantages, could not find the global minimum on many LLS in an efficient enough manner. Our other survey paper [50] examined how complex LLS can be traversed effectively by using biologically inspired meta-heauristic algorithms such as continuous ant colony optimization (ACO), i.e. CACO algorithms. However, our research paper on our novel LLS optimization algorithm, Brownian bridge - CACO (BB-CACO) [49] demostrated that CACO itself, whilst it has many beneficial meta-heuristic features that can traverse LLS, it ultimately could not scale effectively, requiring exponential increases in compute resources. That paper also proposed the BB-CACO algorithm, which mimics CACO but uses Brownian bridge paths to replace the ant agents. Whilst BB-CACO overcame the scaling issues in CACO, it was designed to survey n-dimensional flat surfaces (i.e. certain types of manifolds) embedded in $\mathbb{R}^{n+1}$ space[1]. Whilst BB-CACO overcame the exploration-exploitation dilemma, and it was proven to work in high-dimensional surfaces, albeit flat n-dimensional surfaces, more research work was still required.

In this paper, we wish to extend BB-CACO into a more general and powerful "Itô bridge"[2] CACO (IB-CACO) algorithm. We deliberately adopt this terminology, because we need to cater for a more complex stochastic process than Brownian motion, B_t which can only be scaled by a constant diffusion term σ, giving $dX_t = \sigma\, dB_t$. Here, we will employ Itô processes, which also cater for a drift term μ, giving rise to the more general $dX_t = \mu\, dt + \sigma\, dB_t$. As we demonstrate later in this paper, the drift term serves to guide the trajectory toward more feasible regions characterized by lower loss values, in conjunction with the diffusion term. Since these systems are designed to be adaptive, the parameters must vary over time; hence, more sophisticated Itô stochastic frameworks have been adopted.

We wish to extend BB-CACO from a powerful search algorithm in flat Euclidian space $\mathbb{R}^n$ to LLS, which are non-flat surfaces, so that IB-CACO can be an even more powerful solver or optimizer algorithm. The enhancements that we will make in IB-CACO will overcome the following complexities that are not adequately handled in BB-CACO.

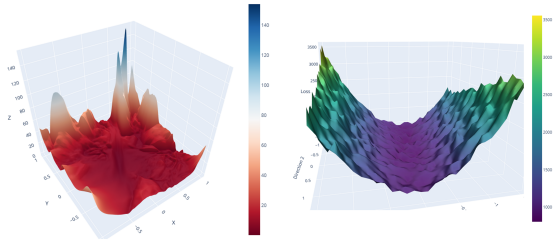
1) Generalizability into LLS.
2) Dynamic nature of LLS.
3) Interior and exterior of LLS.
4) Traversal avoidance dilemma (TAD) of LLS.

A. Generalizability into LLS

We investigate the intricate relationship between an n-dimensional agent traversing a surface in $\mathbb{R}^n$ and the sur-

[1]An n-dimensional flat manifold embedded in $\mathbb{R}^{n+k}$ (for any $k \geq 1$) is called an n-dimensional affine subspace of $\mathbb{R}^{n+k}$ if it is flat and without curvature.

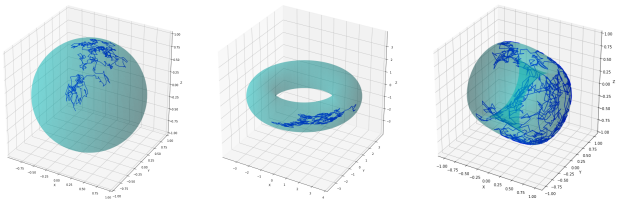
[2]Our use of the term "Itô bridge" is not the first [16], but even more popular terms other than Brownian bridge" are "diffusion bridge" [5], [26], [52] and "conditioned diffusion" [4], [32].



(a). (b).

Fig. 2: Actual LLS from ResNet56 and AlexNet ANNs
(a). A 3-dimensional slice from the ResNet56 LLS.
(b). A 3-dimensional slice from the AlexNet LLS.

rounding geometric variations. For instance, although a 10-dimensional IB-CACO process can evolve within a space of equal or higher dimensionality, there is no assurance that its trajectories will remain confined to the loss surface of interest. Without appropriate constraints, the process may wander into regions of the parameter space that do not meaningfully contribute to identifying the global minimum. To further illustrate these ideas, we apply simple Brownian motion to several low-dimensional manifolds, as shown in Figure 1.



(a). (b). (c).

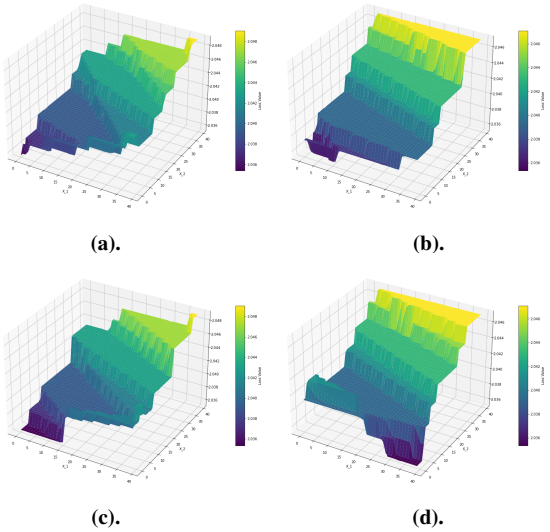
Fig. 1: Brownian Motion on Simple Manifolds
(a). Sphere manifold.
(b). Torus manifold.
(c). Calabi-Yau manifold.

Figure 1 shows that Brownian motion (and indeed Brownian bridges, BB-CACO) can be confined to the LLS manifold, but these examples are too contrived, because in practice, the analytic form of the underlying LLS is rarely available, particularly for modern deep networks with billions of parameters. Moreover, attempts to approximate high-dimensional LLS with simple analytic forms (e.g., polynomial bases) are computationally infeasible in practice, as shown in [17].

Some actual examples of LLS arise from ResNet-56 and AlexNet ANNs, and 3-dimensional slices of these high-dimensional surfaces are shown in Figure 2.

Figure 2 shows the many local minima and nonconvex nature of actual LLS. Interactive visualizations of such slices are available online, for example at:

- https://losslandscape.com/explorer
- https://www.telesens.co/loss-landscape-viz/viewer.html



(a). (b). (c). (d).

Fig. 3: 4 instances of the MNIST LLS for 2 Parameters
(a). LLS Instance 1: Semi-discrete surface with small region for minima.
(b). LLS Instance 2: Semi-discrete surface with slightly larger region for minima.
(c). LLS Instance 3: Semi-discrete surface with even larger region for minima.
(d). LLS Instance 4: Semi-discrete surface with a 'ridge' or well, where a SGD algorithm can become 'stuck'.

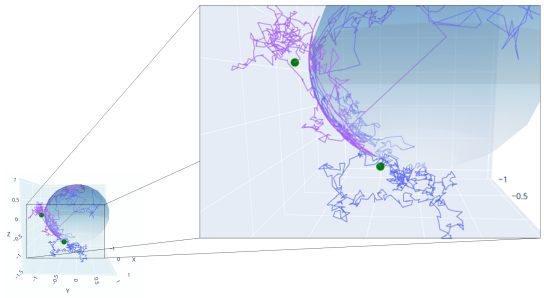
B. Dynamic Nature of LLS

A further complication arises from the inherently dynamic nature of LLS, which evolve throughout the training process as models update their parameters and assimilate new data. This dynamism results from multiple factors, including data variability as models encounter novel patterns [21], continuous parameter updates during optimization, and hyperparameter adjustments that influence how models traverse the loss landscape [13].

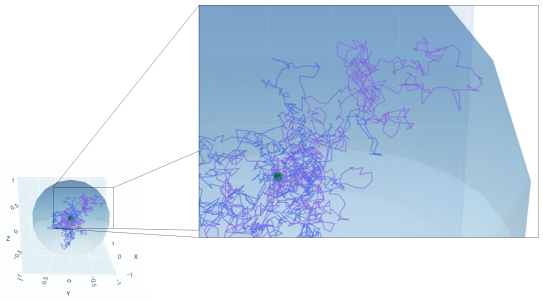
Modeling these dynamic landscapes poses substantial challenges. The process is computationally intensive, necessitating continual monitoring and recalculation of the loss surface. The stochastic nature of training introduces unpredictable fluctuations in the LLS. Moreover, the intricate interplay among data distribution, model architecture, optimization algorithms, and hyperparameters demands sophisticated real-time adaptation mechanisms, rendering dynamic modeling considerably more difficult than in static LLS settings [29].

To illustrate and support this, we take the MNIST dataset and model it via an ANN. The library can create 2D approximations of the loss-landscape surface by selecting two random direction vectors in parameter space and calculating loss values across the resulting space, as shown in Figure 3.

Figure 3 shows 4 different instances of the MNIST LSS for only two parameters, showing the dynamic nature of how the ANN learns over time and adjusts the LLS accordingly. It also shows how the LLS changes due to new data being presented to the ANN, changed and removed from the ANN's scope, over time.



(a).



(b).

Fig. 4: Exterior and Interior of Manifolds

Minor deviations from the surface are tolerable, but beyond a certain threshold, the path agent becomes increasingly unlikely to return within a reasonable timeframe.
(a). In this scenario, the Brownian motion is supposed to traverse the surface on the outside, but ends up moving off the surface and into the exterior of the surface.
(b). In this scenario, the Brownian motion is supposed to traverse the surface on the inside, but ends up moving off the surface and into the interior of the surface.

C. Interior and Exterior of LLS

When modeling Brownian motion on a high-dimensional surface, the process can easily deviate from the surface, making it advantageous to constrain the motion so that it does not drift into irrelevant regions of the surrounding space. The simplest illustration of this arises in the case of three-dimensional Brownian motion constrained to a flat two-dimensional plane. Such constraints are particularly important for LLS, as both the search space and the global minimum reside on the surface itself. This requires careful consideration of several factors, such as dimensional embedding, as some surfaces require embedding in higher dimensions[3] to avoid self-intersections. Furthermore, the motion must adhere to the surface's geometry using, manifold-specific techniques and projections, and surface normal vectors for constraint calculations.

By generalising Brownian motion to stochastic differential equations (SDEs), that incorporate drift and diffusion parameters that can vary over time, one can control the path of motion to respect the surface constraints, as shown in Figure 4.

[3]An n-dimensional surface does not automatically embed in $\mathbb{R}^{n+1}$ and may require $\mathbb{R}^{n+k}$, for $n, k \in \mathbb{N}$.

D. Traversal Avoidance Dilemma (TAD) of LLS

A well-known challenge in ACO, particularly in CACO, is the so-called exploration-exploitation dilemma (EED) [50]. Although this issue originates in the study of collective foraging, it translates naturally to LLS optimization: algorithms must balance refining search within promising local regions against exploring more distant areas where better minima may lie. In addition to the classical EED that underpins most adaptive optimization strategies, we identify a distinct computational trade-off that emerges in iterative search algorithms, particularly those based on spatial expansion mechanisms such as CACO. We refer to this as the Traversal Avoidance Dilemma.

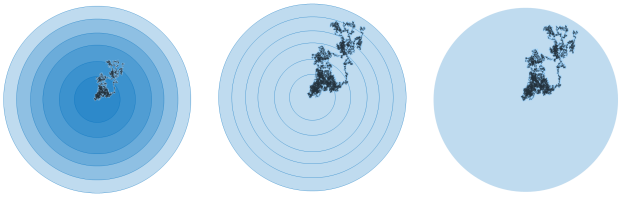
Definition I.1. ***(Traversal Avoidance Dilemma (TAD)):*** *The TAD describes the trade-off between:*

- ***Redundant traversal:*** *The inefficiency incurred when previously explored regions are revisited due to incremental or overlapping expansion of the search space.*
- ***Overextended initialization:*** *The computational burden and potential ineffectiveness of initializing the search over a disproportionately large space, much of which may be irrelevant or contain low-reward regions.*

■

TAD not only arises in algorithms such as BB-CACO and IB-CACO, but also in algorithms where the definition and evolution of the search region are not fixed, but instead grow or adapt over time. At one extreme, starting with a small region and expanding it iteratively can lead to multiple passes over the same terrain. At the other extreme, initializing with an overly large search space may overwhelm the algorithm with noise or dilute its sampling density, especially in high-dimensional scenarios.

While the EED concerns the behavioral balance between refining known optima and discovering new regions, TAD pertains to the *structural efficiency* of the search process –specifically, how the domain is spatially covered over time. This dilemma highlights the impact of different region-expansion strategies on search efficiency, and is illustrated conceptually in Figure 5.



(a). **(b).** **(c).**

Fig. 5: Computational Considerations in the Traversal Avoidance Dilemma

(a). At one extreme lies an approach in which the ant agents explore a relatively small search region that is iteratively expanded over time. While this strategy ensures a focused initial search, it results in significant redundancy –regions are revisited multiple times as the search area grows incrementally.
(b). An effective solution would ideally balance these two extremes –retaining efficiency while ensuring adequate exploration coverage in the TAD (without covering the same region multiple times).
(c). At the other extreme, a large search space is predefined from the outset and surveyed directly by the ant agents. Although this avoids redundant traversal, it introduces two major challenges: (i) the computational cost may become prohibitive, and (ii) it is often unclear how to determine an appropriate size for the search space a priori.

Figure 5 shows that algorithms that fail to account for TAD may exhibit good local behavior (in terms of EED) but still suffer from global inefficiency due to poor traversal planning.

We propose that future designs of CACO-based algorithms should incorporate TAD-aware strategies, such as memory-based exclusion of previously explored regions, hierarchical spatial partitioning, or adaptive boundary growth that avoids redundant overlap.

II. Literature Review

SGD remains a core optimization method in machine learning, but it suffers from noisy convergence [6], [38], sensitivity to hyperparameters [3], vulnerability to local minima and saddle points in nonconvex landscapes [10], slow progress near optima [7], and instability on nonsmooth loss surfaces [47]. These limitations motivate the exploration of stochastic geometrical structures as alternative optimization tools.

Brownian bridges [30] offer a natural model for constrained stochastic trajectories and form a conceptual foundation for more flexible CACO designs. Loss landscapes (LLS), while typically fixed for a dataset, can change indirectly through training dynamics [15], dataset shifts [39], regularization [31], learning rate schedules [15], or the choice of optimization algorithm [24]. Such changes are characteristic of streaming data systems [35], recommendation environments [12], and time-varying forecasting tasks [34].

Traditional gradient-based methods struggle in these dynamic conditions [21], [29]. Several adaptations have been proposed—learning-rate schedules and warm restarts [28], [43], adaptive optimizers such as Adam and RMSprop [19], [51], online or mini-batch updates [9], [46], regularization [20], [23], ensemble methods [40], meta-learning [22], and periodic model refresh strategies [23], [36]. While helpful, these remain fundamentally gradient-driven and thus still inherit core SGD limitations.

Brownian and Itó bridges provide an alternative. These conditioned diffusions —used in ecological modeling [14], [18], [42] —combine local stochasticity with global constraints. Unlike Brownian bridges, Itô bridges have nonconstant drift and diffusion, enabling guided exploratory behaviour. Their theoretical properties have been explored in [1], yet they have not been applied to ACO. Recent ACO improvements employ -greedy schemes and Lévy flights [27], which expand exploration but depend heavily on heavy-tailed distributions whose extreme samples are rare. In contrast, IB-ACO exploits the structure of Itô diffusions to overcome these limitations and support more reliable long-range exploration.

III. Methodology

This section builds upon the mathematical foundations of our earlier BB-CACO framework [49], extending them to the Itô bridge formulation required for IB-CACO.

Required state variables at iteration t,

- $X(t) \in \mathbb{R}^d$: Current position.
- $X(*) \in \mathbb{R}^d$: Best position found (global pheromone).
- $f(*) = \min_{s \le t} f(X(s))$: Best objective value.
- $T(t) \in [0, 1]$: Temperature (exploration control).
- $\sigma(t) > 0$: Exploration scale.
- $\alpha(t) > 0$: Learning rate.
- β: Bridge exploration component.
- $\gamma \in (0, 1)$: Exploration/exploitation proportion.
- δ: Variance reduction function.
- $\hat{\nabla}(t) \in \mathbb{R}^d$: Cached gradient estimate.

At iteration t, the position update equation is,

$$X(t+1) = X(t) + \Delta X_{\text{Explore}}(t) + \Delta X_{\text{Exploit}}(t) \tag{1}$$

where,

$$\Delta X_{\text{Explore}}(t) = \alpha(t)\Big[\underbrace{\sigma(t)T(t)s(t)\boldsymbol{B}_H(t)}_{\text{fBm}} + \underbrace{\mathcal{I}(t)}_{\text{Itô Bridge}} + \underbrace{\mathcal{L}(t)}_{\text{Lévy Flights}} + \underbrace{\mathcal{G}(t)}_{\text{Gravity Sampling}}\Big] \tag{2}$$

and,

$$\Delta X_{\text{Exploit}}(t) = -\alpha(t) \cdot w_g(t) \cdot \hat{\nabla}_{\text{SPSA}}(t) \tag{3}$$

and $w_g(t) = 1 - \gamma \cdot T(t)$. It is the gradual interplay between (2) and (3) that allows IB-CACO to better balance the exploration-exploitation dilema,

- When $T = 1$ (early): $w_g = 0.3$ (30% weight on gradient),
- When $T = 0.5$ (mid): $w_g = 0.65$ (65% weight on gradient),
- When $T = 0$ (late): $w_g = 1.0$ (100% weight on gradient).

Component Definitions:

1) **fBm**:
The use of fractional Brownian motion (fBm) allows for generalised stochastic paths for the CACO agents, via,

$$\boldsymbol{B}_H(t) = \left[B_H^{(1)}(t_n) - B_H^{(1)}(t_0), \cdots, B_H^{(d)}(t_n) - B_H^{(d)}(t_0)\right]^\top$$

where $B_H(i)$ is fBm, B signifies Brownian motion, we have set the Hurst exponent $H = 0.75$ and generated $n = 1,000$ steps.

2) **Itô Bridge**
We adopt a multi-dimensional Brownian bridge [30], [2], but generalize the diffusion term, forming a less well known Itô bridge [5], [11], [33], [41], from being the constant σ to be variable over state and time $\sigma(X_t, t)$, in vector form,

$$dX_t = \frac{x^* - X_t}{T - t}\, dt + \sigma(X_t, t)\, dW_t$$

or in discrete form (Euler step, step Δt):

$$X_{t+\Delta t} = X_t + \frac{x^* - X_t}{T - t}\, \Delta t + \sigma\sqrt{\Delta t}\, \xi_t, \quad \xi_t \sim \mathcal{N}(0, I)$$

where $\frac{x^* - X_t}{T-t}$ is the drift term, $\sigma(X_t, t)$ is the diffusion term, x^* is the target destination and W_t is the continuous Wiener process or simple Brownian motion. ξ_t is the discrete noise term —a random vector sampled from $\mathcal{N}(0, I)$ at each time step[4] .

3) **Lévy Flights**
The use of Lévy flights helps ensure that the path agents are less likely to get stuck in local minima, by sampling more extreme directions from the Lévy distribution, which is heavy-tailed and highly skewed.

$$\mathcal{L}^{(t)} = \begin{cases} \mathbf{c} \cdot \sigma^{(t)} \cdot 0.5, & \text{with probability} \quad p_L \cdot T^{(t)} \\ \mathbf{0}, & \text{otherwise} \end{cases}$$

where $\mathbf{c} \sim \text{Cauchy}(0, 1)^d$ clipped to $[-10\sigma, 10\sigma]$ and $p_L = 0.1$.

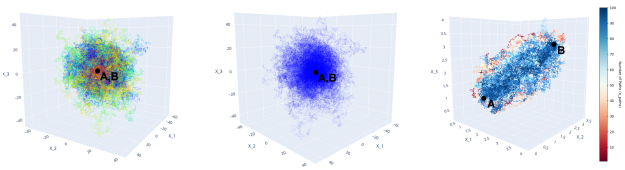
4) **Gravity Sampling**
Another component is the gravity sampling, which works similar to gradient descent, but doesn't involve expensive gradient calculations, and instead samples multiple paths and 'gravitates' towards promissing regions,

$$\mathcal{G}(t) = \gamma \cdot \frac{1}{N_s} \sum_{j=1}^{N_s} w_j(t) \cdot u_j$$

where,

$$w_j(t) = \begin{cases} \frac{f\left(X(t)\right) - f\left(X(t) + \delta \mathbf{u}_j\right)}{|f\left(X(t)\right)| + \epsilon}, & \text{if } f\left(X(t) + \delta \mathbf{u}_j\right) < f\left(X(t)\right) \\ -0.1, & \text{otherwise} \end{cases}$$

[4] $\xi_t \sim \mathcal{N}(0, I) \iff \xi_t = [\xi_{t,1}, \xi_{t,2}, \cdots, \xi_{t,d}]^\top$, where each $\xi_{t,i} \sim \mathcal{N}(0, 1)$ and they are independent.



(a). **(b).** **(c).**

Fig. 6: Brownian Balls and Bridges in 3-Dimensions
(a). 100 simulations of Brownian bridges tethered to start at A and finish at B where $A = B = 0$ (i.e. the origin), with each path having a different colour.
(b). 100 simulations of Brownian bridges tethered to start and finish at the origin, with each path having the same blue colour with opacity, to further highlight the ball nature.
(c). 100 simulations of Brownian bridges tethered to start at arbitrary point A and finish at arbitrary point B (i.e. $A \neq B$), with each path having a Red-to-Blue colour map. Now, the ball symmetry is reduced to a 'prolate spheroid'.

with $\delta = 0.3\sigma(t)$, $N_s = 8$, $\gamma = 0.4$.

5) **Gradient Estimate (SPSA)**
The Simultaneous Perturbation Stochastic Approximation (SPSA) algorithm provides an efficient way to estimate gradients using only two function evaluations per iteration, regardless of the parameter dimension.

$$\hat{\nabla}_{\text{SPSA}}(t) = \begin{cases} \frac{f\left(X(t) + \epsilon(t)\Delta(t)\right) - f\left(X(t) - \epsilon(t)\Delta(t)\right)}{2\epsilon(t)} \cdot \Delta(t), & \text{if } t \mod \text{freq}(T(t)) = 0 \\ \hat{\nabla}_{\text{SPSA}}(t-1), & \text{otherwise (cached)} \end{cases}$$

with adaptive $\epsilon(t)$ and Rademacher perturbation, $\Delta(t) \sim \{-1, +1\}^d$ "uniformly".
After computation, gradient clipping is applied,

$$\hat{\nabla}_{\text{SPSA}}(t) \leftarrow \begin{cases} \hat{\nabla}_{\text{SPSA}}(t), & \text{if } \|\hat{\nabla}_{\text{SPSA}}(t)\|_2 \leq \theta_{\max} \\ \theta_{\max} \cdot \frac{\hat{\nabla}(t)}{\|\hat{\nabla}_{\text{SPSA}}(t)\|_2}, & \text{otherwise} \end{cases}$$

with $\theta_{\max} = 5$.

The algorithm relies entirely on,

- Long-range exploration (fBm).
- Memory of success (bridge to $X(*)$).
- Escape mechanism (Lévy jumps).
- Landscape awareness (gravity via function sampling).
- Low-cost adaptive gradients (SPSA discovery).

IV. Implementation

Extending such Brownian bridges to 3-dimensions, we note that the symmetry of the paths on a flat disk extend to a ball, as shown in Figure 6.

Figure 6 shows how we finally have a viable alternative for ant foraging in continuous ACO algorithms. Not only does it satisfy the requirement of foraging in 2-Dimensions, but also of returning to the origin.

Next, we define a further set of requirements that we would like to see our proposed novel IB-CACO algorithm to have. The relevance of Itô bridges to LLS optimization lies in their ability to constrain stochastic search paths between prescribed start and target regions, while maintaining exploration through

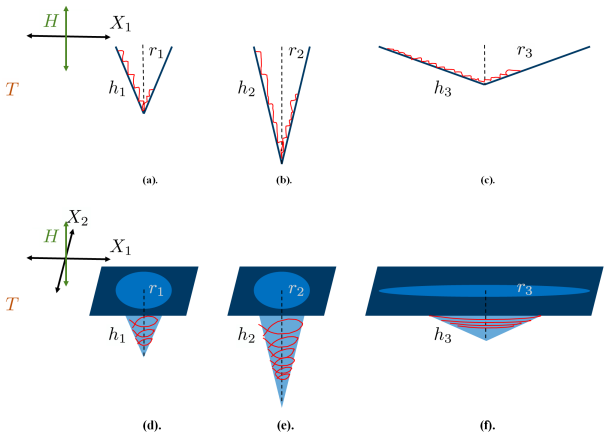


Fig. 7: Traversal Requirements Along Local and Global Minima in Higher Dimensions

1-dimensional Itô Diffusions
A 1-dimensional Itô diffusion moves up and down along the X_1-axis. We notice that we have 3 axes, X_1, height H, and a time axis T that is implied but has not been depicted here.
(a). The minima are traversed up and down the X_1-axis, which is mapped onto the 1-dimensional manifold embedded in $\mathbb{R}^2$.
(b). This minima is lower than the other two minima, making it the global minimum on the manifold. Since $h_2 \sim 2h_1$, in theory the Itô diffusion should usually take longer to find the minimum of this well, since it needs to traverse more of the manifold.
(c). This minima has half the height but twice the width (or radius). It should thus take just as long to traverse the same distance as in (b).
2-dimensional Itô Diffusions
A 2-dimensional Itô diffusion moves up and down along the X_1 and X_2-axis. We notice that we have 4 axes, X_1, X_2, height H, and a time axis T that is implied but has not been depicted here.
(d). The minima is traversed up and down the X_1-axis and X_2-axis, which is mapped onto the 2-dimensional manifold embedded in $\mathbb{R}^3$.
(e). $r_2 = r_1/2$ yet $h_2 = 2h_1$, this means that we would expect the paths to take twice as long to find the minimum.
(f). Since $h_3 \sim h_1/2$, in theory the Itô diffusion should usually take longer to find the minimum of this well, since it needs to traverse more of the manifold.

diffusion. This structure mitigates the tendency of algorithms such as SGD and its variants to become trapped in local minima, since the bridge dynamics impose a global directional bias toward target minima. Unlike momentum-based methods, which require sensitive hyperparameter tuning, bridge-based dynamics inherently balance exploration and convergence without relying on heuristic adjustments. We will address this in the Methodology (§III), but for now, we will simply illustrate the functional requirements that one would expect from such an algorithm, as shown in Figure 7.

Figure 7 shows that unlike the SGD with Momentum (SGD-M), we would need to have some more general mechanism, such as a meta-heuristic 'gravity' penalty function, that does not need to be hyperparameter tuned. Preliminary experiments (see Results §V) suggest that bridge-based optimization achieves more consistent convergence across diverse landscapes, with reduced sensitivity to hyperparameters compared to SGD-M. While further benchmarking is required, these results highlight the potential of IB-CACO as a robust alternative. It also shows that the steepness of the well would dictate different values for the momentum, making the use of it more problematic.

To highlight the corresponding differences and benefits of the IB-CACO over CACO, the IB-CACO pseudocode for surfaces is outlined in Algorithm 1.

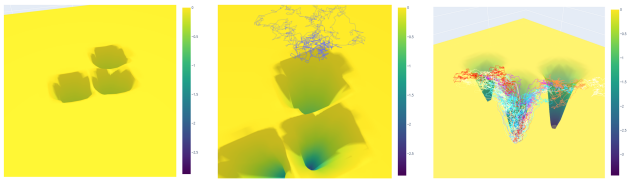
Algorithm 1 IB-CACO

Input: f, x_0, $T_{\max}$, H, σ_0, β, γ, α_0
Output: Approximate global minimum x^*

$x, x^*, f^* \leftarrow x_0, x_0, f(x_0)$; $T, \sigma, \alpha \leftarrow 1, \sigma_0, \alpha_0$; $\nabla f \leftarrow 0_d$; $n_{\text{stag}} \leftarrow 0$
for $t = 1$ **to** $T_{\max}$ **do**
  // --- Adaptive SPSA Gradient (2 Evaluations Periodically) ---
  if $t \bmod (5 - 3T) < 1$ **then**
    $\varepsilon \leftarrow \max(10^{-6}, 0.1^{1-T}\sigma)$; $\Delta \leftarrow \text{sign}(\text{randn}(d))$; $\nabla f \leftarrow \text{clip}\left(\frac{f(x+\varepsilon\Delta)-f(x-\varepsilon\Delta)}{2\varepsilon}\Delta, 5\right)$
  // --- Exploration Components (No Evaluations) ---
  bridge $\leftarrow \beta T \frac{x^*-x}{\|x^*-x\|}$; levy $\leftarrow$ (rand $< 0.1T$) clip(Cauchy$(d), -10\sigma, 10\sigma)\sigma/2$; fbm $\leftarrow$ fBM(d, H)
  // --- Gravity term (Sample-Averaged Directionality) ---
  $g \leftarrow 0_d$; $u_j \leftarrow \frac{\text{randn}(N_s,d)}{\|\text{randn}(N_s,d)\|}$; **foreach** u_j **do**
    $f_j \leftarrow f(x + 0.3\sigma u_j)$; $g \leftarrow g + \gamma\,\text{sign}(f(x) - f_j)u_j$
  $g \leftarrow g/N_s$
  // --- Combined Update ---
  scale $\leftarrow$ multi_scale$(t, T_{\max})$; $x_c \leftarrow x + \alpha[\sigma T \text{scale}(\text{fbm} + \text{bridge} + \text{levy} + g) - (1 - 0.7T)\nabla f]$
  // --- Evaluation & Acceptance ---
  $f_c \leftarrow f(x_c)$; **if** $f_c < f(x)$ ***or*** *rand()* $< e^{-(f_c - f(x))/(T\|f(x)\|)}$ **then**
    $x \leftarrow x_c$; **if** $f_c < f^*$ **then**
      $x^* \leftarrow x_c$, $f^* \leftarrow f_c$, $T \leftarrow \min(1.15T, 1)$, $n_{\text{stag}} \leftarrow 0$
    else
      $n_{\text{stag}} + +$
  else
    $n_{\text{stag}} + +$
  // --- Parameter Decay & Restart ---
  $T \leftarrow \max(0.98^{1+(n_{\text{stag}}>5)}T, 0.01)$; $\sigma \leftarrow \max(0.998\sigma, 0.01)$; $\alpha \leftarrow \max(0.999\alpha, 0.001)$;
  if $n_{stag} > 20 \wedge T < 0.1 \wedge$ *rand()* < 0.3 **then**
    $x \leftarrow x^* + \mathcal{N}(0, 5\sigma, d)$, $T \leftarrow 0.5$, $n_{\text{stag}} \leftarrow 0$
  if *converged* **then**
    break
return x^*

Algorithm 1 was applied over three key surfaces, which are all flat, as a preliminary investigation before analysing the Results of the Algorithm on more complex surfaces, as shown in Figure 8.

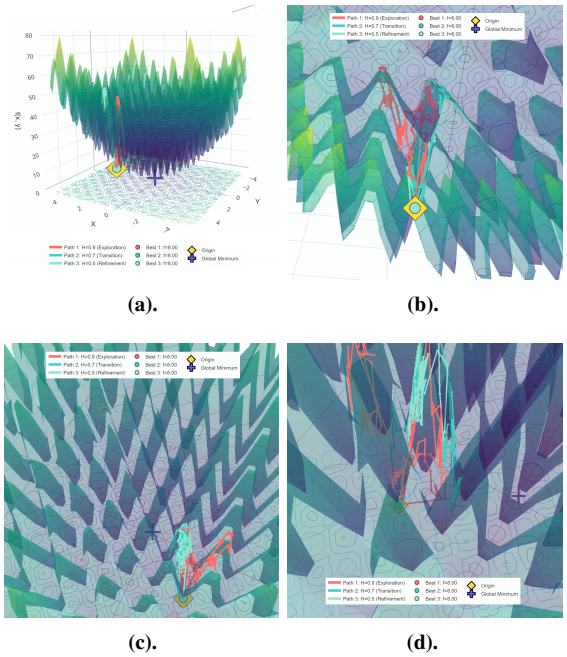
Figure 8 suggests that SGD and BB-CACO should be ruled out from this paper, as one can not undertake a like-for-like comparison between these algorithms and IB-ACO. The Results section will thus involve three surfaces, two which SGD can not effectively find the global minimum, and one



(a). **(b).** **(c).**

Fig. 8: Preliminary Implementation Results

10 randomized starting points on this surface for all 3 algorithms.
(a). SGD did not produce any paths since the surface is flat and so SGD could not descend and hence move.
(b). BB-CACO produced paths but whilst they could move on flat surfaces, they could not descend as they are 2-dimensional in nature. A 'plain vanilla' 3-dimensional Brownian motion would not have any way to constrain itself to the surface, unless it was a Brownian bridge process, and even then it would require additional parameters that are present in IB-CACO.
(c). IB-CACO is able to effectively descend down and ascend up minima, to find the global minimum in a reasonable time.



(a). **(b).** **(c).** **(d).**

Fig. 9: IB-CACO Starting Off Rastragin Surface

(a). IB-CACO randomly starts below the Rastragin surface.
(b). Zooming in, we see how the start and end of each path is the same, i.e. a bridge.
(c). Looking from the top down, we better see how IB-CACO can traverse down *and up* local minima, unlike gradient based algorithms.
(d). Zooming in further still, we can see additional details of the bridge mechanism.

surface where SGD can be compared to IB-CACO.

V. Results

A. Experiment 1 – Bridge Ascent Results

Figure 9 illustrates how the initial bridge-constrained exploration behaves on the Rastragin surface and highlights the effects of fBm-driven path memory during early optimisation.

Figure 9 demonstrates how the choice of Hurst exponent H shapes the memory and directional bias of each path. Higher values of H promote long-range correlation and encourage persistent exploratory trajectories, while lower values revert the search to standard Brownian motion. The staged reduction from $H = 0.9$ to $H = 0.5$ therefore provides a natural coarse-to-fine search schedule in which early exploration identifies promising regions and later rounds progressively refine the estimate of the global minimum.

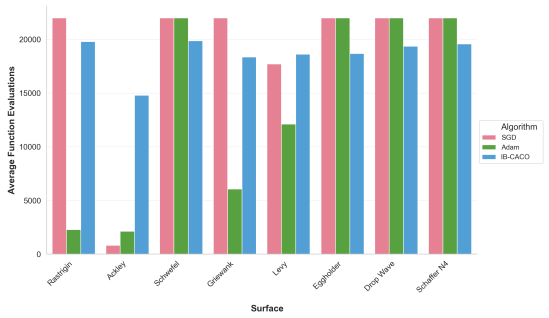


Fig. 10: Computational Efficiency Across Benchmark Surfaces

IB-CACO achieved lower losses in fewer steps or iterations than the other algorithms over the majority of surfaces (i.e. Schwefel, Eggholder, Drop Wave, Shaffer N4) and came close second best on other surfaces (i.e. Rastragin, Griewank).

B. Experiment 2 – Benchmark Surfaces Results

The relative efficiency of IB-CACO across a broad suite of benchmark surfaces is summarised in Figure 10, which compares the number of steps required by each algorithm to reach competitive loss values.

Figure 10 highlights the consistent performance gains of IB-CACO on challenging, highly non-convex landscapes. These trends are reflected quantitatively in Table I, which reports mean, median, standard deviation, and average function evaluations across all benchmark functions.

Table I confirms the visual trends seen in Figure 10. IB-CACO consistently achieves strong performance on surfaces where traditional gradient-based optimisers struggle, particularly on highly multimodal and discontinuous landscapes such as Ackley, Levy, Styblinski–Tang, and several of the rugged test functions (Eggholder, Drop Wave, Schaffer N.4). In cases where gradient-based algorithms already perform well (e.g., Griewank, Michalewicz), IB-CACO remains competitive while requiring comparable or fewer iterations. Overall, IB-CACO demonstrates favourable trade-offs between search efficiency and solution quality across diverse landscape geometries.

C. Experiment 3 - Complex Manifold Results

We first apply IB-CACO to our complex loss surface, defined over a two-dimensional parameter space. Starting at a location where the SGD algorithm would descend into one of the first two local minima and become trapped, failing to reach the global minimum. In our earlier experiments, BB-CACO was developed as a variant of the standard CACO framework, but

TABLE I: Average Performance and Evaluations Across (AE) Benchmark Functions

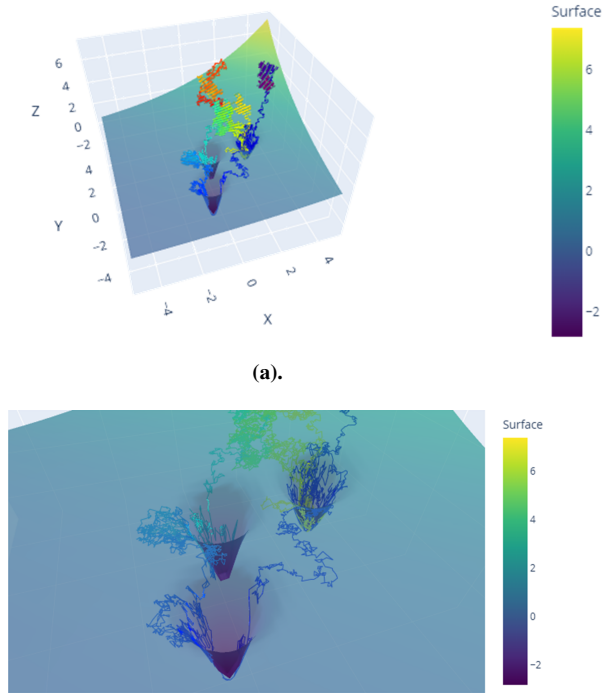
Surface	Algorithm	Mean	Median	Std	AE
Rastrigin	SGD	44.66	45.06	15.53	22001
	Adam	44.11	44.77	15.54	2291
	Efficient IB-CACO	**28.84**	27.07	16.63	19786
Ackley	SGD	8.71	8.92	1.31	827
	Adam	8.68	8.92	1.30	2136
	Efficient IB-CACO	**1.11**	0.02	2.19	14800
Schwefel	**SGD**	**1078.65**	1087.43	277.47	22001
	Adam	1610.22	1652.18	323.33	22001
	Efficient IB-CACO	1285.07	1253.06	305.28	19860
Griewank	SGD	0.283	0.032	0.407	22001
	Adam	**0.017**	0.015	0.009	6068
	Efficient IB-CACO	0.023	0.017	0.016	18363
Levy	SGD	3.206	2.885	1.581	17710
	Adam	3.204	2.884	1.582	12116
	Efficient IB-CACO	**1.722**	1.444	1.529	18615
Michalewicz	SGD	-1.691	-1.620	0.845	14431
	Adam	**-2.650**	-2.696	0.851	2187
	Efficient IB-CACO	-1.425	-1.371	0.899	6043
Styblinski–Tang	SGD	-164.73	-167.56	15.23	900
	Adam	-164.73	-167.56	15.23	3546
	Efficient IB-CACO	**-168.50**	-167.56	13.62	19851
Eggholder	**SGD**	**-298.81**	-281.43	115.32	22001
	Adam	-197.35	-190.26	86.47	22001
	Efficient IB-CACO	-296.65	-284.35	133.47	18678
Drop Wave	**SGD**	**1.021**	0.651	1.137	22001
	Adam	567.84	535.83	325.66	22001
	Efficient IB-CACO	103.13	0.000	154.77	19355
Schaffer N.4	**SGD**	**1.642**	1.165	1.257	22001
	Adam	568.29	536.32	325.65	22001
	Efficient IB-CACO	86.37	0.501	154.35	19568

we observed that it failed to explore loss landscape (LLS) structures with deep wells effectively. To address this, we modified BB-CACO to better adhere to the surface geometry of the LLS, allowing it to go up and down wells, which is the first modification for BB-CACO to become IB-CACO. This behaviour was achieved through a gravity-inspired penalty term in the objective function: at each step, the algorithm evaluates a penalty value $P \propto \Delta h$, where Δh is the vertical displacement between consecutive steps. Downward steps reduce the penalty, while upward steps increase it. This biases the search trajectory toward descending paths, while still allowing escape when necessary. This also spawns multiple paths in parallel and finds the global minimum quickly, as shown in Figure 11.

Figure 11 illustrates the impact of incorporating the gravity penalty into IB-CACO. The modified trajectories demonstrate improved exploration of the LLS, enabling the algorithm to escape shallow local minima and approach the global minimum more effectively, as shown in Figure 12.

Figure 12 demonstrates the success of IB-CACO over traditional SGD-related optimization algorithms on a highly nonconvex surface, our exponential well surface. We wish to further test IB-CACO superiority over another highly nonconvex surface, our 'top hat' surface, as shown in Figure 13.

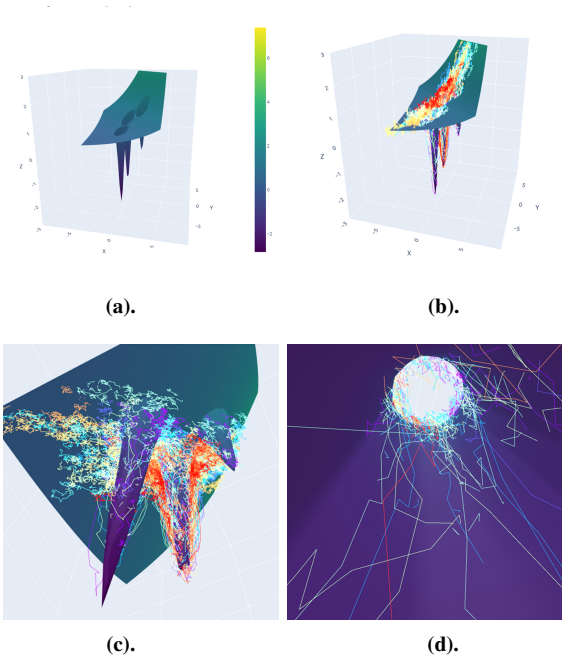
Figure 13 further clearly demonstrates IB-CACO's superior abilities to find the global minimum where other algorithms fail. But how does it perform on less theoretical surfaces, i.e. on actual LLS? Now turning to the Modified national institute of standards and technology (MNIST) ANN dataset, the Python library [25] can be used to produce 2-dimensional approximations of the LLS topology around a point in param-



(a).

(b).

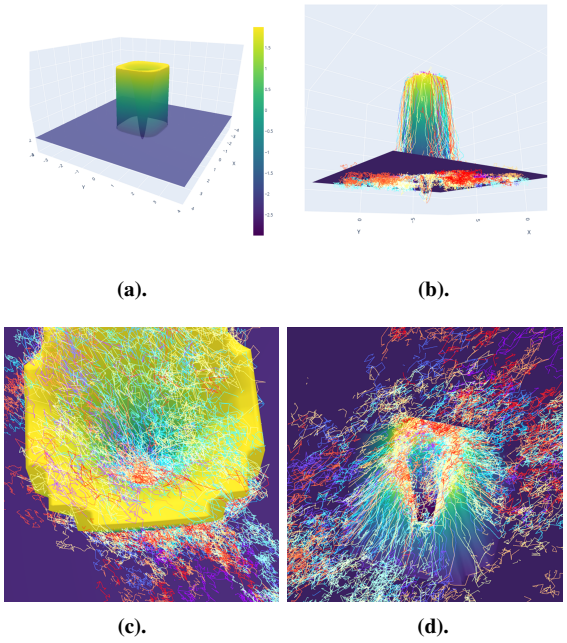
Fig. 11: IB-CACO on our Complex Manifold I
(a). Multiple IB-CACO paths.
(b). One path magnified to traverse down and up wells to ultimately find the global minimum.



(a). **(b).**

(c). **(d).**

Fig. 12: IB-CACO on our Complex Manifold II
(a). Tripple exponential well surface,
$f(x,y) = \exp\left(0.2x + 0.2y\right) - 2\exp\left(-\frac{(x-2)^2+y^2}{0.2}\right) - 2.5\exp\left(-\frac{(x+0.2)(x-0.2)+y^2}{0.2}\right) - 3.5\exp\left(-\frac{(x+0.5)^2+(y+2)^2}{0.2}\right)$.
(b). 10 IB-CACO paths descend and ascend on the surface.
(c). Underneath the surface, we see that IB-CACO has found the global minimum.
(d). Entering the opening of the global minimum, the paths descend and after finding it, continue to ascend it.



(a). **(b).**

(c). **(d).**

Fig. 13: IB-CACO on our Top Hat Surface

(a). Another highly nonconvex surface, our 'top hat' surface, defined as, $f(x, y) = 2\tanh(2 - (x^4 + y^4)) - 5\exp(-(x^2 + y^2)/0.2)$. Notice that to find the global minimum, the only an algorithm that can climb will find it.
(b). The IB-CACO paths start on the flat plane, a place where SGD algorithms would stop descending. It then climbs the 'head' part of the 'top hat'.
(c). IB-CACO then descends into the 'head' of the 'hat'.
(d). Looking underneath the 'hat', we see that IB-CACO has indeed found the global minimum, below the flat plane, and in the middle of the hat.

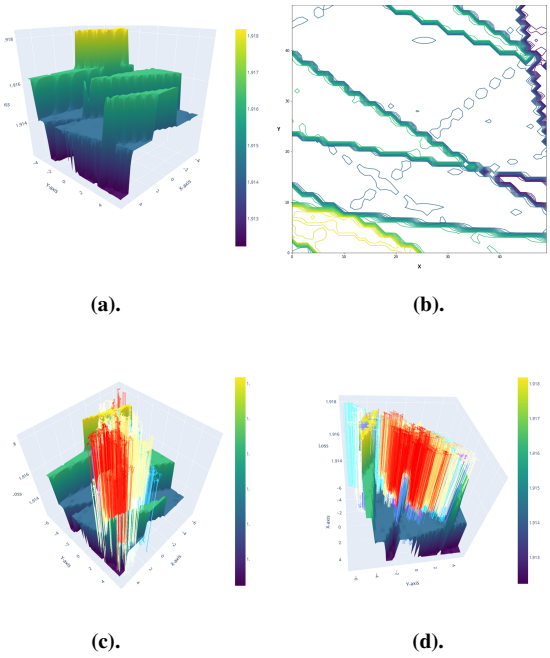
eter space. This is achieved by sampling two random direction vectors in parameter space, and computing the loss at a number of points on the plane defined by the two vectors, as shown in Figure 14.

Figure 14(c) and (d) shows that IB-CACO also works well on real life LLS from ANNs. (c) and (d) are two separate IB-CACO instances on the same LLS instance and both quickly found the global minimum on the LLS.

VI. Discussion and Interpretation of Results

A. Experiment 1 - Discussion and Interpretation of Bridge Ascent Results

The bridge-ascent results demonstrate that, unlike gradient-based optimisers —which must initialise on or above the surface —IB-CACO can begin from any point in the vicinity of the landscape. Despite the extreme multimodality of the Rastragin function, the algorithm consistently navigated its dense local minima and rugged topology. While these findings were encouraging, a broader and more systematic evaluation was necessary. Accordingly, we conducted additional experiments across a wide range of benchmark surfaces and employed a more rigorous data-extraction methodology to assess performance comprehensively.



(a). **(b).**

(c). **(d).**

Fig. 14: IB-CACO Over the MNIST Surface

(a). 3D plot of two random parameters from the MNIST LLS.
(b). Contour plot of two random parameters from the MNIST LLS.
(c). IB-CACO applied over the LLS finds the global minimum.
(d). IB-CACO applied over the LLS rotated 45.

TABLE II: Summary of Best Algorithm per Surface

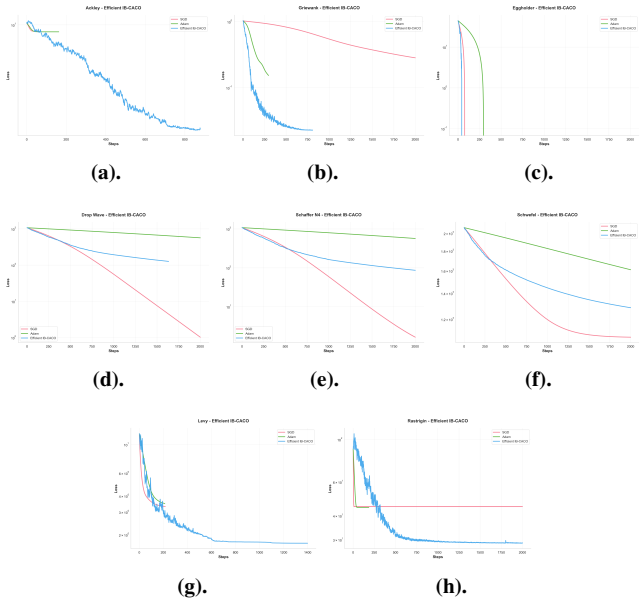
Surface	Winner	Remarks
Rastrigin	IB-CACO	Strong improvement in mean loss
Ackley	IB-CACO	Large performance gap vs Adam/SGD
Schwefel	SGD	Better accuracy despite higher evals
Griewank	Adam	Lowest mean, highly efficient
Levy	IB-CACO	46% lower mean vs Adam
Michalewicz	Adam	Most efficient convergence
Styblinski–Tang	IB-CACO	Slightly better optimum
Eggholder	SGD	Retains best mean solution
Drop Wave	SGD	Outperforms hybrids on mean
Schaffer N.4	SGD	Most stable performance
Overall Wins	**IB-CACO: 4**	Adam: 3, SGD: 3 (out of 10)

B. Experiment 2 - Discussion and Interpretation of Benchmark Surfaces Results

Clear differences in convergence behaviour were evidenced among SGD, Adam, and IB-CACO across a diverse set of non-convex benchmark landscapes, and the loss rates over time are shown in Figure 15.

Figure 15 shows that Efficient IB-CACO often achieves lower final losses than SGD and Adam on highly multimodal surfaces, and does so while maintaining stable, monotonic descent on functions where gradient-based methods stall or diverge. These trends correspond directly to the performance rankings reported in Table II.

Table II shows the experimental comparison across ten benchmark functions provides a clear picture of each algorithm's strengths. The Efficient IB-CACO hybrid method outperformed both traditional gradient descent (SGD) and adaptive gradient optimization (Adam) on 4 out of 10 surfaces, particularly excelling on multimodal and rugged landscapes such as Rastrigin, Ackley, Levy, and Styblinski–Tang. These



(a). (b). (c).

(d). (e). (f).

(g). (h).

Fig. 15: Optimization Algorithm Comparison - Complex Surfaces
(a). Ackley surface.
(b). Griewank surface.
(c). Eggholder surface
(d). Drop wave surface
(e). Schaffer N.4 surface
(f). Schwefel surface
(g). Levy Surface
(h). Rastrigin surface

surfaces are characterized by multiple local minima, where exploration–exploitation balance is critical. The algorithm's design —emphasizing adaptive frequency of gradient use, gradient recycling, and fractional weighting —contributed to its ability to escape local minima and converge toward globally superior regions.

In contrast, Adam showed the strongest performance on smoother, differentiable landscapes such as Griewank and Michalewicz, where fast, stable convergence is favored over broad exploration. Its computational efficiency was also evident, typically requiring fewer evaluations than both SGD and IB-CACO. SGD, while comparatively inefficient, retained competitive accuracy in highly deceptive functions like Schwefel, Eggholder, and Schaffer N.4, suggesting its resilience in environments where overadaptation can lead to premature convergence.

From an efficiency perspective, Efficient IB-CACO demonstrated up to 70–90% reduction in gradient computations through its stochastic perturbation (SPSA) approach and adaptive gradient application. Despite requiring more total evaluations per iteration in some cases, its overall resource utilization remained competitive due to fewer function calls per optimization cycle. The statistical analysis (ANOVA results with p-values < 0.001 for most functions) confirms that these observed performance differences are statistically significant rather than due to random variation.

In conclusion, the Efficient IB-CACO algorithm shows a compelling trade-off: while not universally dominant, it consistently provides robust, statistically significant improvements on complex, multimodal surfaces where conventional gradient-based optimizers struggle. Its philosophy of treating the gradient as a bias rather than a core driver of optimization is empirically validated, offering a promising direction for hybrid metaheuristic-gradient methods.

C. Experiment 3 - Discussion and Interpretation of Complex Manifold Results

To further broaden the evaluation, two additional low-dimensional yet highly challenging manifolds were constructed in three-dimensional space: a complex manifold featuring two local minima and one global minimum, and a top-hat manifold whose global minimum is effectively unreachable for any gradient-based method, as doing so would require ascending the outer "hat" structure. These experiments reinforced the earlier findings regarding IB-CACO's effectiveness; statistical comparison tables were unnecessary because gradient-based optimizers fail outright on these deliberately adversarial surfaces.

VII. Conclusions

The IB-CACO algorithm outperforms traditional optimizers in several key areas, including in high-dimensional optimization problems. It excels on non-convex surfaces with multiple local minima, demonstrates an improved capability to escape saddle points, and enables more efficient exploration of complex loss landscapes. Additionally, its adaptive step sizing adjusts to the surface's geometry, and its temperature-based exploration gradually narrows in on promising regions —a behavior that scales effectively even in high-dimensional loss landscapes. Although Adam and RMSProp may be faster initially, IB-CACO generally requires fewer steps to locate the global minimum, particularly in complex and high-dimensional landscapes. IB-CACO has the ability to compliment existing solvers and yet offers new avenues for future research.

Funding Information

The first author was supported by an Australian Government Research Training Program (RTP) Scholarship.

Author's Contributions

Aldo Taranto: Conceptualization, methodology, software, investigation, writing - original draft, writing - review and editing, formal analysis and visualization.
Ron Addie: Validation, writing - review and editing.
Bernardo P. Nunes: Validation, writing - review and editing.

Ethics

This article is original and contains unpublished material. The corresponding author confirms that all of the other authors have read and approved the manuscript and that there are no ethical issues involved.

References

[1] Alsolami, A., Burridge, J., and Gnacik, M. Size and shape of tracked Brownian bridges. *Journal of Physics A: Mathematical and Theoretical 53*, 26 (2020), 265001.

[2] Beghin, L., and Orsingher, E. On the maximum of the generalized Brownian bridge. *Lithuanian Mathematical Journal 39* (1999), 157–167. https://www.researchgate.net/profile/Enzo-Orsingher/publication/236984395_On_the_maximum_of_the_generalized_Brownian_bridge/links/02e7e51aee2ba07219000000/On-the-maximum-of-the-generalized-Brownian-bridge.pdf.

[3] Bengio, Y. Practical recommendations for gradient-based training of deep architectures. *Neural Networks: Tricks of the Trade 7700* (2012), 437–478.

[4] Beskos, A., Papaspiliopoulos, O., Roberts, G. O., and Fearnhead, P. Exact and computationally efficient likelihood-based estimation for discretely observed diffusion processes. *Journal of the Royal Statistical Society: Series B 68*, 3 (2006), 333–382.

[5] Bladt, M., and Sørensen, M. Simple simulation of diffusion bridges with application to likelihood inference for diffusions. *Bernoulli 20*, 2 (2014), 645–675.

[6] Bottou, L. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*. Springer, 2010, pp. 177–186.

[7] Bottou, L., Curtis, F. E., and Nocedal, J. Optimization methods for large-scale machine learning. *SIAM Review 60*, 2 (2018), 223–311.

[8] Boyd, S., and Vandenberghe, L. *Convex optimization*. Cambridge University Press, 2004. https://xiesaining.wordpress.com/wp-content/uploads/2012/09/mlss2011_vandenberghe_convex.pdf.

[9] Cesa-Bianchi, N., and Lugosi, G. Prediction, learning, and games. In *Online Learning and Prediction* (2006). Covers theoretical foundations of online learning / incremental updates.

[10] Dauphin, Y. N., Pascanu, R., Gulcehre, C., Cho, K., Ganguli, S., and Bengio, Y. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in Neural Information Processing Systems* (2014), pp. 2933–2941.

[11] D'Auria, B., and Ferriero, A. Optimal stopping times for a class of itô diffusion bridges. *arXiv preprint arXiv:1909.02916* (2019). Preprint.

[12] Ebrahimi, A., Zhang, Z., and Stefanidis, K. Dynamic user preferences optimization in time-aware recommendations. *DARLIA P (conference / workshop)* (2025). Models how user preferences evolve over time in recommender systems.

[13] Eslami, M., Eramian, H., Gameiro, M., Kalies, W., and Mischaikow, K. Extracting global dynamics of loss landscape in deep learning models. *arXiv arXiv:2106.07683* (2021), 1–17. https://arxiv.org/pdf/2106.07683.

[14] Fisher, J. W., Walter, W. D., and Avery, M. L. Brownian bridge movement models to characterize birds' home ranges. *Condor 115* (2013), 298–305.

[15] Fort, S., and Jastrzebski, S. Large scale structure of neural network loss landscapes. In *Advances in Neural Information Processing Systems (NeurIPS)* (2019), vol. 32.

[16] Guidoum, A. C. Sim.diffproc: Diffusion processes simulation. https://cran.r-project.org/web/packages/Sim.DiffProc/, 2023. Vignette containing explicit examples labelled "Itô Bridge".

[17] Har-Peled, S., and Varadarajan, K. High-dimensional shape fitting in linear time. *Proceedings of the nineteenth annual symposium on Computational geometry* (2003), 39–47. https://dl.acm.org/doi/pdf/10.1145/777792.777799.

[18] Horne, J. S., Garton, E. O., Krone, S. M., and Lewis, J. S. Analyzing animal movements using Brownian bridges. *Ecology 88* (2007), 2354–2363.

[19] Kingma, D. P., and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014). A widely used adaptive optimizer based on first and second moment estimates.

[20] Krogh, A., and Hertz, J. A. A simple weight decay can improve generalization. In *Advances in Neural Information Processing Systems* (1992), vol. 4, p. 950–957. One of the earlier analyses of L2 regularization ("weight decay") in neural nets.

[21] Lavin, E., and Ruiz-Garcia, M. Dynamical loss functions shape landscape topography and improve learning in artificial neural networks. *arXiv arXiv:2410.10690* (2024), 1–6. https://arxiv.org/pdf/2410.10690.

[22] Li, D., Yang, Y., Song, Y., and Hospedales, T. M. Learning to generalize: Meta-learning for domain generalization. *arXiv preprint arXiv:1710.03463* (2017). Proposes simulating domain shifts during meta-training to learn robustness to novel domains.

[23] Li, e. a. Domain generalization through meta-learning: A survey. *arXiv preprint arXiv:2404.02785* (2024). Surveys meta-learning methods for adapting to domain shifts (i.e. continuously evolving distributions).

[24] Li, H., et al. Loss landscapes and optimization in over-parameterized non-linear models. *Journal of Complexity (or corresponding journal from link)* (2021).

[25] Li, H., Xu, Z., Taylor, G., Studer, C., and Goldstein, T. Visualizing the loss landscape of neural nets. *Advances in neural information processing systems (NIPS) 31* (2018), 1–11. https://proceedings.neurips.cc/paper_files/paper/2018/file/a41b3bb3e6b050b6c9067c67f663b915-Paper.pdf.

[26] Lindström, E. A regularized bridge approach to stochastic differential equations. *Brazilian Journal of Probability and Statistics 21*, 2 (2007), 177–205.

[27] Liu, Y., Cao, B., and Li, H. Improving ant colony optimization algorithm with epsilon greedy and Levy flight. *Complex Intelligent Systems 7* (2021), 1711–1722. https://doi.org/10.1007/s40747-020-00138-3.

[28] Loshchilov, I., and Hutter, F. Sgdr: Stochastic gradient descent with warm restarts. In *ICLR (workshop / conference)* (2017). Introduces warm restarts (cosine annealing with periodic resets) to escape local minima.

[29] Nakhodnov, M., Kodryan, M., Lobacheva, E., and *et al.* Loss function dynamics and landscape for deep neural networks trained with quadratic loss. *Dokl. Math. 106*, 1 (2022), S43–S62. https://doi.org/10.1134/S1064562422060187.

[30] Øksendal, B. *Stochastic Differential Equations: An Introduction with Applications*. Springer, New York, 1995.

[31] or authors from preprint, A. Cgd: Modifying the loss landscape by gradient regularization. *Preprint / arXiv* (2025).

[32] Pedersen, A. R. A new approach to maximum likelihood estimation for stochastic differential equations based on discrete observations. *Scandinavian Journal of Statistics 22*, 1 (1995), 55–71.

[33] Peluchetti, S. Diffusion bridge mixture transports, schrödinger bridge problems and generative modeling. *Journal of Machine Learning Research 24* (2023), 1–51.

[34] Qin, D., Li, Y., Chen, W., Zhu, Z., Wen, Q., Sun, L., Pinson, P., and Wang, Y. Evolving multi-scale normalization for time series forecasting under distribution shifts. *arXiv preprint* (2024). Addresses distribution shifts in time-series forecasting over time.

[35] Qin, Z., Liang, Z., Xu, L., Wu, W., Wu, M., Shen, W., and Wang, W. Freewayml: An adaptive and stable streaming learning framework for dynamic data streams. *ICDE (conference paper / proceedings)* (2025). Demonstrates continuous adaptation of models to streaming data in financial, network, energy, etc.

[36] Regol, F., Schwinn, L., Sprague, K., Coates, M., and Markovich, T. When to retrain a machine learning model. *arXiv preprint arXiv:2505.14903* (2025). Addresses the decision of when to retrain models under evolving data distributions, i.e. "model refresh.".

[37] Rockafellar, R. Conjugate convex functions in optimal control and the calculus of variations. *Journal of Mathematical Analysis and Applications 32*, 1 (1970), 174–222. https://core.ac.uk/download/pdf/81214013.pdf.

[38] Ruder, S. An overview of gradient descent optimization algorithms. https://arxiv.org/abs/1609.04747, 2016. arXiv:1609.04747.

[39] Ruiz-Garcia, M., Zhang, G., Schoenholz, S. S., and Liu, A. J. Tilting the playing field: Dynamical loss functions for machine learning. *Preprint / arXiv* (2021).

[40] Sakai, S., Tsuge, S., and Hasegawa, T. Noisy deep ensemble: Accelerating deep ensemble learning via noise injection. *arXiv preprint arXiv:2504.05677* (2025). Perturbs a parent model to explore different minima in an ensemble, reducing training time while diversifying model solutions.

[41] Schauer, M., van der Meulen, F., and van Zanten, H. J. Guided proposals for simulating multi-dimensional diffusion bridges. *Bernoulli 23*, 4A (2017), 2104–2130.

[42] SILVA, I., CRANE, M., SUWANWAREE, P., STRINE, C., AND GOODE, M. Using dynamic Brownian bridge movement models to identify home range size and movement patterns in king cobras. *PLoS One 13*, e0203449 (2018).
[43] SMITH, L. N. Cyclical learning rates for training neural networks. *arXiv preprint arXiv:1506.01186* (2015). Proposes letting the learning rate vary cyclically instead of monotonically decreasing; relevant for "learning rate schedules / warm restarts".
[44] SPALL, J. C. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control 37*, 3 (1992), 332–341.
[45] SPALL, J. C. Implementation of the simultaneous perturbation algorithm for stochastic optimization. *IEEE Transactions on Aerospace and Electronic Systems 34*, 3 (1998), 817–823.
[46] SRIVASTAVA, N., HINTON, G., KRIZHEVSKY, A., SUTSKEVER, I., AND SALAKHUTDINOV, R. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research 15*, 56 (2014), 1929–1958. Classic reference for dropout regularization.
[47] SUN, R. G., WANG, Y., AND LIU, H. Optimization for machine learning: theory and algorithms. *Foundations and Trends in Machine Learning 12*, 1-3 (2019), 1–231.
[48] TARANTO, A., AND ADDIE, R. Survey of Loss Lanscape Surfaces: Theory, Applications and Algorithms. *engrXiv* (2025), 1–45. https://engrxiv.org/preprint/view/5069.
[49] TARANTO, A., ADDIE, R., AND NUNES, B. Brownian Bridge - CACO (BB-CACO). *TechRxiv* (2025), 1–31. https://www.techrxiv.org/users/959089/articles/1332208-brownian-bridge-continuous-ant-colony-optimization-bb-caco.
[50] TARANTO, A., NUNES, B., AND ADDIE, R. Survey of Continuous Ant Colony Optimization: Theory, Applications and Algorithms. *Journal of Computer Science* (2025), 1–42. https://www.techrxiv.org/users/959089/articles/1332208/master/file/data/BB-CACO/BB-CACO.pdf.
[51] TIELEMAN, T., AND HINTON, G. Lecture 6.5—rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning (lecture notes)* (2012). Describes RMSProp, another adaptive learning rate method.
[52] WHITAKER, G. A., GOLIGHTLY, A., BOYS, R. J., AND SHERLOCK, C. Improved bridge constructs for stochastic differential equations. *Statistics and Computing 27*, 4 (2017), 885–900.

VIII. APPENDICES

IX. GENERALIZING TO n-DIMENSIONAL SPACE

1) 2D Brownian motion on flat 2D space mapped to walk on a 2D manifold in 3D space: To map a 2-dimensional Brownian motion on a flat 2D space to a walk on a 2D manifold in 3D space, we need to understand a few key concepts; Brownian motion, coordinate mapping, and the nature of the manifold.

Brownian Motion in 2D Space:
In a flat 2-dimensional (2D) space, Brownian motion can be described by the following SDEs,

$$dX_1(t) = dW_1(t),$$
$$dX_2(t) = dW_2(t),$$

where $\Big(X_1(t), X_2(t)\Big)$ are the coordinates in the 2D space at time t, and $W_1(t), W_2(t)$ are independent Wiener processes (standard Brownian motions) in the X_1 and X_2 directions, respectively.

Mapping to a 2D Manifold in 3D Space:
To map this motion to a 2-dimensional manifold embedded in a 3-dimensional (3D) space, we need to define a mapping from the flat 2D coordinates (X_1, X_2) to the 3D coordinates (X_1, X_2, X_3) on the manifold. Let $\Phi : \mathbb{R}^2 \to \mathbb{R}^3$ be this mapping such that,

$$\Phi(X_1, X_2) = \Big(\phi_1(X_1, X_2),\ \phi_2(X_1, X_2),\ \phi_3(X_1, X_2)\Big).$$

Geodesic Motion on the Manifold:
The Brownian motion on the manifold can be described by geodesic paths on the manifold. The SDEs on the manifold are given by,

$$d\phi_1(t) = dW_{\phi_1}(t),$$
$$d\phi_2(t) = dW_{\phi_2}(t),$$
$$d\phi_3(t) = dW_{\phi_3}(t),$$

where $\Big(\phi_1(t),\ \phi_2(t),\ \phi_3(t)\Big)$ are the coordinates on the manifold at time t.

2) nD Brownian motion on flat nD space mapped to walk on a nD manifold in $(n+1)$-D space: **Brownian Motion in n-D Space**

In a flat n-dimensional (nD) space, Brownian motion is described by the following SDEs,

$$dX_1(t) = dW_1(t),$$
$$\cdots$$
$$dX_n(t) = dW_n(t),$$

where $\Big(X_1(t), \cdots, X_n(t)\Big)$ are the coordinates in 3D space at time t, and $W_1(t), \cdots, W_n(t)$ are independent Wiener processes in the $X_1, \cdots, X_n$ directions, respectively.

Mapping to a nD Manifold in $(n+1)$D Space

To map this motion to a n-dimensional manifold embedded in a $(n+1)$-dimensional ($(n+1)$D) space, we need to define a mapping from the flat nD coordinates $(X_1, \cdots, X_n)$ to the $(n+1)$D coordinates $(\phi_1, \cdots, \phi_n, \phi_{n+1})$ on the manifold. Let $\Phi : \mathbb{R}^n \to \mathbb{R}^{n+1}$ be this mapping such that,

$$\Phi(X_1, \cdots, X_n) = \Big(\phi_1(X_1, \cdots, X_n), \cdots, \phi_n(X_1, \cdots, X_n),\ \phi_{n+1}(X_1, \cdots, X_n)\Big).$$

Geodesic Motion on the Manifold

The Brownian motion on the manifold can be described by geodesic paths on the manifold. The SDEs on the manifold are given by,

$$d\phi_1(t) = dW_1(t),$$
$$\cdots$$
$$d\phi_n(t) = dW_n(t),$$

$$d\phi_{n+1}(t) = dW_{n+1}(t),$$

where $\Big(\phi_1(t), \cdots, \phi_n(t), \phi_{n+1}(t)\Big)$ are the coordinates on the manifold at time t.

X. Nonconvexity and Convexity in n-Dimensional Space

A function in one dimension $f = (x_1)^2$ is convex (i.e. a parabola), and another function in two dimensions $f = (x_1)^2 + (x_2)^2$ is convex (i.e. a paraboloid), and so an n-dimensional version, $f = (x_1)^2 + \cdots + (x_n)^2$ is also convex. These functions belong to a family of functions known as *quadratic forms*, which are convex in any number of dimensions. A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if its Hessian matrix (the matrix of second-order partial derivatives) is positive semi-definite. For the function $f(x_1, x_2, \cdots, x_n) = x_1^2 + x_2^2 + \cdots + x_n^2$, the Hessian matrix is, $\nabla^2 f = 2I$ where, I is the identity matrix. The identity matrix is positive definite, and scaling it by 2 (or any positive constant) doesn't change this property, ensuring that $\nabla^2 f$ remains positive definite. This implies that the function is convex.

For a function in the form $f(x_1, x_2, \cdots, x_n) = \sum_{i=1}^{n} = c_i x_i^2$ where, $c_i > 0$, the function is convex regardless of the number of dimensions, n.

(Linear Combinations Involving Only Convex Funcions) [8], [37]: If a function $f : \mathbb{R}^n \to \mathbb{R}$ is twice differentiable and its Hessian matrix, $\nabla^2 f(x)$ is positive semi-definite for all $x \in \mathbb{R}^n$, then f is a convex function. ■

An application of this is that the function $f(x_1, x_2, \cdots, x_n) = \sum_{i=1}^{n} x_i^2$, the Hessian matrix is $2I$, which is positive definite, thus proving that this function is convex.

(Linear Combinations Involving Some Non-Convex Funcions) [8], [37]: If $f(x)$ is a non-convex function, then any linear combination of $f(x)$ and other functions that include the non-convex function $f(x)$ will *generally* be non-convex. ■

If $f(x_3)$ is a non-convex function, then a function like $g(x_1, x_2, x_3) = \alpha f(x_3) + \beta h(x_1, x_2)$, where α and β are constants and h is any other function, will *typically* be non-convex due to the non-convex nature of $f(x_3)$. Consider $f(x) = -x^2$ which is a non-convex function (a downward opening parabola). If we construct a higher-dimensional function like $g(x_1, x_2) = -x_1^2 + x_2^2$, then the Hessian matrix of g will have both positive and negative eigenvalues due to the terms $-x_1^2$ and x_2^2. This mixed Hessian indicates that $g(x_1, x_2)$ is non-convex.

We want to find an example where $f(x_1)$ is non-convex, and a linear combination with another function $g(x_2)$ results in a convex function. Consider the following functions in two dimensions,

$$f(x_1) = -x_1^2 \text{ (non-convex)}, \quad g(x_2) = x_2^2 \text{ (convex)}.$$

Let's form a linear combination,

$$h(x_1, x_2) = \alpha f(x_1) + \beta g(x_2).$$

Choosing $\alpha = 1$, $\beta = 2$. Therefore,

$$h(x_1, x_2) = f(x_1) + 2g(x_2) = -x_1^2 + 2x_2^2.$$

To determine whether $h(x_1, x_2)$ is convex, we check its Hessian matrix H,

$$H = \begin{bmatrix} \frac{\partial^2 h}{\partial x_1^2} & \frac{\partial^2 h}{\partial x_1 \partial x_2} \\ \frac{\partial^2 h}{\partial x_1 \partial x_2} & \frac{\partial^2 h}{\partial x_2^2} \end{bmatrix} = \begin{bmatrix} -2 & 0 \\ 0 & 4 \end{bmatrix}$$

The eigenvalues of this Hessian matrix are -2 and 4. Since the Hessian matrix has both positive and negative eigenvalues, the function,

$$h(x_1, x_2) = -x_1^2 + 2x_2^2$$

is a saddle point function, which is neither entirely convex nor concave.

Convex Combination Leading to Convexity: To achieve a convex result, we need to ensure all second derivatives are non-negative. Consider,

$$f(x_1) = x_1^2 - 4x_1 \text{ (non-convex, shifted parabola)},$$

$$g(x_2) = x_2^2 \text{ (convex)}.$$

Let's form a linear combination, $h(x_1, x_2) = \alpha f(x_1) + \beta g(x_2)$, where $\alpha = 1/4$, $\beta = 1$.

Therefore,

$$h(x_1, x_2) = \frac{1}{4}(x_1^2 - 4x_1) + x_2^2 = \frac{1}{4}x_1^2 - x_1 + x_2^2.$$

To determine whether $h(x_1, x_2)$ is convex, we check its Hessian matrix H,

$$H = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & 2 \end{bmatrix}.$$

The eigenvalues of this Hessian matrix are $1/2$ and 2, which are both positive. The function $h(x_1, x_2) = \frac{1}{4}x_1^2 - x_1 + x_2^2$ is convex despite $f(x_1)$ being non-convex. This example shows that with appropriate weighting, a linear combination of non-convex and convex functions can result in an overall convex function.

A. Simultaneous Perturbation Stochastic Approximation (SPSA)

Let $f(\theta)$ denote the objective function to be minimized, where $\theta = [\theta_1, \theta_2, \ldots, \theta_d]^\top \in \mathbb{R}^d$. At iteration t, the SPSA gradient estimate is obtained as follows.

1) **Generate random perturbations:**

$$\Delta_{\text{SPSA}}(t) = \big[\Delta_1(t), \Delta_2(t), \ldots, \Delta_d(t)\big]^\top,$$

where each $\Delta_i(t)$ is drawn independently from a symmetric Bernoulli distribution where,

$$\mathbb{P}\big(\Delta_i(t) = +1\big) = \mathbb{P}\big(\Delta_i(t) = -1\big) = \tfrac{1}{2}.$$

2) **Evaluate perturbed losses:**

$$y_+ = f\big(\theta(t) + c_t\Delta(t)\big), \qquad y_- = f\big(\theta(t) - c_t\Delta(t)\big),$$

where $c_t > 0$ is a small perturbation constant.

3) **Compute the simultaneous perturbation gradient estimate:**

$$\hat{\boldsymbol{g}}(t) = \frac{y_+ - y_-}{2c_t} \begin{bmatrix} \frac{1}{\Delta_1(t)} \\ \frac{1}{\Delta_2(t)} \\ \vdots \\ \frac{1}{\Delta_d(t)} \end{bmatrix} = \frac{y_+ - y_-}{2c_t}\, \Delta(t)^{-1},$$

where $\Delta(t)^{-1}$ denotes the elementwise reciprocal of $\Delta(t)$.

4) **Update rule:**

$$\theta(t+1) = \theta(t) - a_t\, \hat{\boldsymbol{g}}(t),$$

where $a_t > 0$ is the learning rate at iteration t.

The SPSA estimator requires only two function evaluations per iteration, independent of the parameter dimension d, making it highly efficient for high-dimensional and noisy optimization problems [44], [45].

References

[1] Alsolami, A., Burridge, J., and Gnacik, M. Size and shape of tracked Brownian bridges. *Journal of Physics A: Mathematical and Theoretical 53*, 26 (2020), 265001.

[2] Beghin, L., and Orsingher, E. On the maximum of the generalized Brownian bridge. *Lithuanian Mathematical Journal 39* (1999), 157–167. https://www.researchgate.net/profile/Enzo-Orsingher/publication/236984395_On_the_maximum_of_the_generalized_Brownian_bridge/links/02e7e51aee2ba07219000000/On-the-maximum-of-the-generalized-Brownian-bridge.pdf.

[3] Bengio, Y. Practical recommendations for gradient-based training of deep architectures. *Neural Networks: Tricks of the Trade 7700* (2012), 437–478.

[4] Beskos, A., Papaspiliopoulos, O., Roberts, G. O., and Fearnhead, P. Exact and computationally efficient likelihood-based estimation for discretely observed diffusion processes. *Journal of the Royal Statistical Society: Series B 68*, 3 (2006), 333–382.

[5] Bladt, M., and Sørensen, M. Simple simulation of diffusion bridges with application to likelihood inference for diffusions. *Bernoulli 20*, 2 (2014), 645–675.

[6] Bottou, L. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*. Springer, 2010, pp. 177–186.

[7] Bottou, L., Curtis, F. E., and Nocedal, J. Optimization methods for large-scale machine learning. *SIAM Review 60*, 2 (2018), 223–311.

[8] Boyd, S., and Vandenberghe, L. *Convex optimization*. Cambridge University Press, 2004. https://xiesaining.wordpress.com/wp-content/uploads/2012/09/mlss2011_vandenberghe_convex.pdf.

[9] Cesa-Bianchi, N., and Lugosi, G. Prediction, learning, and games. In *Online Learning and Prediction* (2006). Covers theoretical foundations of online learning / incremental updates.

[10] Dauphin, Y. N., Pascanu, R., Gulcehre, C., Cho, K., Ganguli, S., and Bengio, Y. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in Neural Information Processing Systems* (2014), pp. 2933–2941.

[11] D'Auria, B., and Ferriero, A. Optimal stopping times for a class of itô diffusion bridges. *arXiv preprint arXiv:1909.02916* (2019). Preprint.

[12] Ebrahimi, A., Zhang, Z., and Stefanidis, K. Dynamic user preferences optimization in time-aware recommendations. *DARLIA P (conference / workshop)* (2025). Models how user preferences evolve over time in recommender systems.

[13] Eslami, M., Eramian, H., Gameiro, M., Kalies, W., and Mischaikow, K. Extracting global dynamics of loss landscape in deep learning models. *arXiv arXiv:2106.07683* (2021), 1–17. https://arxiv.org/pdf/2106.07683.

[14] Fisher, J. W., Walter, W. D., and Avery, M. L. Brownian bridge movement models to characterize birds' home ranges. *Condor 115* (2013), 298–305.

[15] Fort, S., and Jastrzebski, S. Large scale structure of neural network loss landscapes. In *Advances in Neural Information Processing Systems (NeurIPS)* (2019), vol. 32.

[16] Guidoum, A. C. Sim.diffproc: Diffusion processes simulation. https://cran.r-project.org/web/packages/Sim.DiffProc/, 2023. Vignette containing explicit examples labelled "Itô Bridge".

[17] Har-Peled, S., and Varadarajan, K. High-dimensional shape fitting in linear time. *Proceedings of the nineteenth annual symposium on Computational geometry* (2003), 39–47. https://dl.acm.org/doi/pdf/10.1145/777792.777799.

[18] Horne, J. S., Garton, E. O., Krone, S. M., and Lewis, J. S. Analyzing animal movements using Brownian bridges. *Ecology 88* (2007), 2354–2363.

[19] Kingma, D. P., and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014). A widely used adaptive optimizer based on first and second moment estimates.

[20] Krogh, A., and Hertz, J. A. A simple weight decay can improve generalization. In *Advances in Neural Information Processing Systems* (1992), vol. 4, p. 950–957. One of the earlier analyses of L2 regularization ("weight decay") in neural nets.

[21] Lavin, E., and Ruiz-Garcia, M. Dynamical loss functions shape landscape topography and improve learning in artificial neural networks. *arXiv arXiv:2410.10690* (2024), 1–6. https://arxiv.org/pdf/2410.10690.

[22] Li, D., Yang, Y., Song, Y., and Hospedales, T. M. Learning to generalize: Meta-learning for domain generalization. *arXiv preprint arXiv:1710.03463* (2017). Proposes simulating domain shifts during meta-training to learn robustness to novel domains.

[23] Li, E. A. Domain generalization through meta-learning: A survey. *arXiv preprint arXiv:2404.02785* (2024). Surveys meta-learning methods for adapting to domain shifts (i.e. continuously evolving distributions).

[24] Li, H., et al. Loss landscapes and optimization in over-parameterized non-linear models. *Journal of Complexity (or corresponding journal from link)* (2021).

[25] Li, H., Xu, Z., Taylor, G., Studer, C., and Goldstein, T. Visualizing the loss landscape of neural nets. *Advances in neural information processing systems (NIPS) 31* (2018), 1–11. https://proceedings.neurips.cc/paper_files/paper/2018/file/a41b3bb3e6b050b6c9067c67f663b915-Paper.pdf.

[26] Lindström, E. A regularized bridge approach to stochastic differential equations. *Brazilian Journal of Probability and Statistics 21*, 2 (2007), 177–205.

[27] Liu, Y., Cao, B., and Li, H. Improving ant colony optimization algorithm with epsilon greedy and Levy flight. *Complex Intelligent Systems 7* (2021), 1711–1722. https://doi.org/10.1007/s40747-020-00138-3.

[28] Loshchilov, I., and Hutter, F. Sgdr: Stochastic gradient descent with warm restarts. In *ICLR (workshop / conference)* (2017). Introduces warm restarts (cosine annealing with periodic resets) to escape local minima.

[29] Nakhodnov, M., Kodryan, M., Lobacheva, E., and *et al.* Loss function dynamics and landscape for deep neural networks trained with quadratic loss. *Dokl. Math. 106*, 1 (2022), S43–S62. https://doi.org/10.1134/S1064562422060187.

[30] Øksendal, B. *Stochastic Differential Equations: An Introduction with Applications*. Springer, New York, 1995.

[31] or authors from preprint, A. Cgd: Modifying the loss landscape by gradient regularization. *Preprint / arXiv* (2025).

[32] Pedersen, A. R. A new approach to maximum likelihood estimation for stochastic differential equations based on discrete observations. *Scandinavian Journal of Statistics 22*, 1 (1995), 55–71.

[33] Peluchetti, S. Diffusion bridge mixture transports, schrödinger bridge problems and generative modeling. *Journal of Machine Learning Research 24* (2023), 1–51.

[34] Qin, D., Li, Y., Chen, W., Zhu, Z., Wen, Q., Sun, L., Pinson, P., and Wang, Y. Evolving multi-scale normalization for time series forecasting under distribution shifts. *arXiv preprint* (2024). Addresses distribution shifts in time-series forecasting over time.

[35] Qin, Z., Liang, Z., Xu, L., Wu, W., Wu, M., Shen, W., and Wang, W. Freewayml: An adaptive and stable streaming learning framework for dynamic data streams. *ICDE (conference paper / proceedings)* (2025). Demonstrates continuous adaptation of models to streaming data in financial, network, energy, etc.

[36] Regol, F., Schwinn, L., Sprague, K., Coates, M., and Markovich, T. When to retrain a machine learning model. *arXiv preprint arXiv:2505.14903* (2025). Addresses the decision of when to retrain models under evolving data distributions, i.e. "model refresh.".

[37] Rockafellar, R. Conjugate convex functions in optimal control and the calculus of variations. *Journal of Mathematical Analysis and Applications 32*, 1 (1970), 174–222. https://core.ac.uk/download/pdf/81214013.pdf.

[38] Ruder, S. An overview of gradient descent optimization algorithms. https://arxiv.org/abs/1609.04747, 2016. arXiv:1609.04747.

[39] Ruiz-Garcia, M., Zhang, G., Schoenholz, S. S., and Liu, A. J. Tilting the playing field: Dynamical loss functions for machine learning. *Preprint / arXiv* (2021).

[40] Sakai, S., Tsuge, S., and Hasegawa, T. Noisy deep ensemble: Accelerating deep ensemble learning via noise injection. *arXiv preprint arXiv:2504.05677* (2025). Perturbs a parent model to explore different minima in an ensemble, reducing training time while diversifying model solutions.

[41] Schauer, M., van der Meulen, F., and van Zanten, H. J. Guided proposals for simulating multi-dimensional diffusion bridges. *Bernoulli 23*, 4A (2017), 2104–2130.

[42] Silva, I., Crane, M., Suwanwaree, P., Strine, C., and Goode, M. Using dynamic Brownian bridge movement models to identify home range size and movement patterns in king cobras. *PLoS One 13*, e0203449 (2018).

[43] Smith, L. N. Cyclical learning rates for training neural networks. *arXiv preprint arXiv:1506.01186* (2015). Proposes letting the learning rate vary cyclically instead of monotonically decreasing; relevant for "learning rate schedules / warm restarts".

[44] Spall, J. C. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control 37*, 3 (1992), 332–341.

[45] Spall, J. C. Implementation of the simultaneous perturbation algorithm for stochastic optimization. *IEEE Transactions on Aerospace and Electronic Systems 34*, 3 (1998), 817–823.

[46] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research 15*, 56 (2014), 1929–1958. Classic reference for dropout regularization.

[47] Sun, R. G., Wang, Y., and Liu, H. Optimization for machine learning: theory and algorithms. *Foundations and Trends in Machine Learning 12*, 1-3 (2019), 1–231.

[48] Taranto, A., and Addie, R. Survey of Loss Lanscape Surfaces: Theory, Applications and Algorithms. *engrXiv* (2025), 1–45. https://engrxiv.org/preprint/view/5069.

[49] Taranto, A., Addie, R., and Nunes, B. Brownian Bridge - CACO (BB-CACO). *TechRxiv* (2025), 1–31. https://www.techrxiv.org/users/959089/articles/1332208-brownian-bridge-continuous-ant-colony-optimization-bb-caco.

[50] Taranto, A., Nunes, B., and Addie, R. Survey of Continuous Ant Colony Optimization: Theory, Applications and Algorithms. *Journal of Computer Science* (2025), 1–42. https://www.techrxiv.org/users/959089/articles/1332208/master/file/data/BB-CACO/BB-CACO.pdf.

[51] Tieleman, T., and Hinton, G. Lecture 6.5—rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning (lecture notes)* (2012). Describes RMSProp, another adaptive learning rate method.

[52] Whitaker, G. A., Golightly, A., Boys, R. J., and Sherlock, C. Improved bridge constructs for stochastic differential equations. *Statistics and Computing 27*, 4 (2017), 885–900.

Dr. Aldo Taranto Aldo is undertaking postdoctoral research at the Australian National University (ANU) in advanced optimization techniques for high-dimensional machine learning, under an Australian Defence innovation scholarship. He is currently Director of AI Research & Development at MetaModelR Corporation. Aldo holds a BSc(Math) from Monash University (1996), GradDipEd from University of Melbourne (1997), MBSys from Monash University (1998), MB(Acc) from RMIT University (2006) and was awarded a PhD(Math) from the University of Southern Queensland (2022), for his research in stochastic differential equations and their application in mathematical finace and algorithmic trading.

A/Prof. Ron Addie Ron is an Adjunct Associate Professor at the University of Southern Queensland (UniSQ). He began his research career at Telstra Research Laboratories, where he completed a PhD in Markov Additive Processes and co-developed virtual paths —now foundational to ATM broadband networks. He also advanced performance models for Gaussian traffic in core networks. Joining UniSQ in 1993, he served as Head of Mathematics and Computing (2004–2006), taught across multiple IT and science programs, and supervised over 10 PhD students. His Netml software supported 1000+ students and 19+ publications over 15 years. Though retired in 2022, he remains active in research and postgraduate supervision.